Original Research Article

Extracting a Bounded Region from a Map Using Flood Fill Algorithm

Abstract: Extracting noise-free images from a bounded region is an important task in image processing. Editing a map and extracting a region fromthe map is challenging. It is useful in some contexts to have a region in a separate sheet. In this image processing, we have used the Flood Fill algorithm to extract a region from the image map. To achieve that goal, we had worked in our study to separate a bounded region on a map. Usually, a scanned map may contain a lot of noisy information. So we have to process the image to remove useless and noisy information from the map. We had quantized the image to a binary one. In the second phase, we have applied a gray color to separate the desired position from a map. Our main objective of the study to extract a bounded region from mapping an image that contains noisy information and removes it. We have experimented with several maps and it works successfully.

Keywords: **Image Processing, Map, Extract, Bounded Region, Flood-Fill Algorithm.**

1. Introduction

1.2 Prelude

Extracting noise-free images isan important topicin computer vision and image processing fields. At present image processing has a very huge appliance and nearly all of the technical fields are dominated by image processing.In[1]an image consists of a two-dimensional array of numbers. The color or gray shade represented for a given picture element (pixel) rely on the number stored in the array for that pixel. The ordinary type of image data is black and white. It is a binary image since each pixel is either 0 or1. The more complex type of image data is gray-scale, where each pixel takes on a value between zero and the number of gray-scales or gray levels that the scanner can record.

These images become visible like similar black-and-white photographs they are black, whiteandshades of gray. Most gray-scale images today have 256 shades of gray. **People can characterize about 40 shades of gray, so a 256-shade image.** The most critical type of image is color. Color images are same to gray-scale except that there are three bands, or channels, same to the colors red, green, and blue. Thus, each pixel has three values combined with it. A color scanner uses red, green, and blue filters to generate those values. Images are available via the Internet, scanners, and digital pictures. Any pictures shown on the internet can be downloaded by pressing the right mouse button when the pointer is on the image. This contracts the image to the user's PC usually in a JPEG format.

In this work, we have used the flood fill algorithm for extracting noise-free bounded regions from the original image map. A scanned map may contain a lot of noisy and useless information. Often the noisy content interspersed with the main content. It’s a great problem to use the map. We have processed the image of the map to remove useless and noisy information from that and extract the bounded region.

1.3 Motivation

The scientific system of imageprocessing is concerned with the knowledgebehind spurious systems that extract significant information from images. Extracting meaningful information from a map is so much interesting task in image processing. The image data can contain many forms, such as video sequences, views from multiple cameras, 2D images,3D scanners, variousmedical scanning devices. The technicalorder of imageprocessing seeks to apply its knowledge and models to the working of computer vision systems.Image processing and image analysis tend to focus on 2D images, how to transform one image to another by pixel-wise operations such as grayscale conversion, contrast enhancement, local operations such as edge extraction or geometrical transformationsor noise removal, or extract meaningful information, and also remove unnecessary information.The characterization signifies that image processing.

1.4 Objective the work

- ✓ To extract the bounded region from a map.
- ✓ To remove unnecessary information from a map.
- ✓ To extract meaningful information from a map.
- ✓ To remove noisy information because the scanned map may contain a lot of noisy information.

2. Related Work

In [2]this work, the researcher compared to the Flood Fill algorithm, the edited Flood-Fill algorithm offers animportant reduction in cell distance updates. With a lesser number of distances to update, the micro mouse which uses the modified flood fill can traverse from cell to cell at a higher speed. In [3]image processing image may contain a lot of noisy information. To avoid this noise, we have applied the flood- filling algorithmthat begins from a known pixel in a boundedregion and recursively identifies all the pixels within the adjacent region of the known pixel. To touch all pixels in each region by a 4-connected or 8-connected pixel a stack may be needed, and pixels within the region will be coming recursively. In [4]computer graphics there are many algorithms to fill the polygon. This work they show the comparative performance both flood-fill and boundary-fill algorithm. We have realized that both algorithms have optimized some real applications. It is suitable for filling large polygon. We also see that flood-fill algorithms represent efficient results than the boundary-fill algorithms in this study.In[5] this study, we have been realized that to find the object using the flood-fill algorithm. Image segmentation is actually used to identify objects and boundaries in images. This paper shows the most identical regions identify using the flood-fill algorithm. The image is first segmented by mean shifted segmentation. The users only require to exactly identify the original features of objects. In[6] this paper, the flood-fill algorithm is used robotics path identification. FF dividing matrices allows and development of path planning of mobile agents. This algorithm creates a short movement that is important to reduce the energy costs by the path of the robot.

3. Methodology

3.1 Introduction

In this chapter, the algorithms and datasets are described. These algorithms are used to extract meaningful information from the map usingthe flood-fill algorithm. The map images of different types are provided as the dataset in these algorithms. **A simple map image is presented below: -**

Fig.1: Sample Mapimage

3.2 Proposed Work

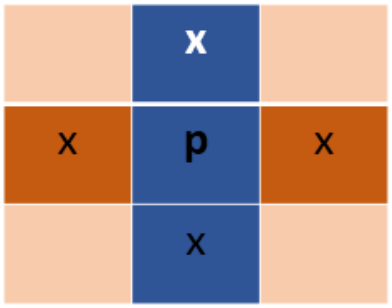
The proposed systemof an algorithm is shown here.To process an image, it is very much important to configure the OpenCV (open-source computer vision library).It can work with Visual C++. It can use the Intel Image Processing libraries (IPL). The configuration process of that interesting library is not so hard.

Algorithm

1 Read images from the dataset.
2 Calculate the height, width, step width, and channel of the stored image.
3 Pre-process images like Make the gray-scale image completely black and white
4 If (RGB< 100)
  Then make the Color White.
  Else
  Make the color black.
5 Apply the FloodFill Algorithm to select the expected region and change the color to black.
6 Set the path where want to save.
7 Exit

3.3 Overview of the Flood Fill Algorithm

In this section, we have described the Flood fill algorithm. It is useful in cases where there no single color boundary for the polygon that is the boundary has multiple colors. In[7]the flood fill algorithm instead of filling color till we appointment a specific boundary color we just fills the pixels with the default color. It is used function to fill connected, same-colored areas with a different color. In the flood-fill algorithm, there are two methods in which the flood-fill algorithm can be implemented these are 4-connected pixel and 8-connected pixel.In this work, we have used the 4-connected pixel method and we check 4 pixels adjacent to the current pixel namely towards the top, bottom, left, right. So we fill the area with the 4-connected pixel method. The working procedure and figure are given below: -



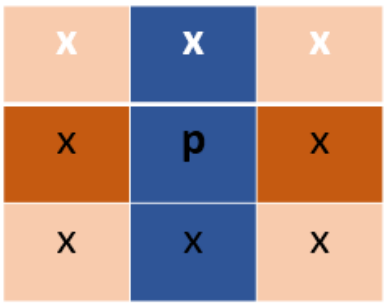


Fig 2:4-Connected Pixel **Fig3: 8-Connected Pixel**

3.3.1 Working procedure:

Step 1: First initialization the 4 values namely x, y, fill-color, and default-color. Where x and y are coordinated positions of the initial interior pixel we start the flood fill algorithm and fill-color is the color we want to fill default-color is the color of the interior pixel.

Step 2: if the color of the node is not equal to the default-color, return.
Step 3: Set the color of the node to the replacement-color.
Step 4: Set the color of the node to the bottom of the node to the replacement-color.
Step 5: Set the color of the node to the top of the node to the replacement-color.
Step 6: Set the color of the node to the left of the node to the replacement-color.
Step 7: Set the color of the node to the right of the node to the replacement-color.
Step 4: Exit.

The flood-fill(FF) also starting with seed within the region. It checks to see if the pixel has the region's original color. If the answer is yes, it fills the pixel with a new color and uses each of the pixels adjacent as a new seed in a recursive call. If the answer is no, it returns to the caller. FF method shares key similarities in its workingrule with the boundary-fill algorithm.

4. Result And Discussion

In this section, we have discussed the result of the task. The target of our experiment is to extract the bounded region we find out the image data and convert the gray-scale image into the black and white images. Experiment results are shown in different steps these are given below.

Step1: At first to display the image data we need an image as an input. This image is as follows:

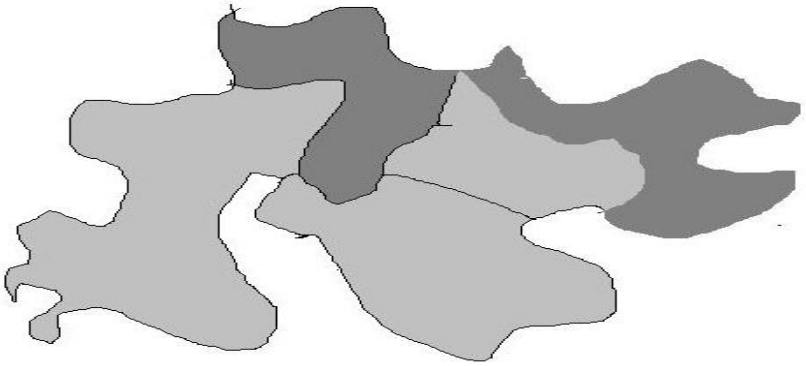


Fig.4: Input image

After processing this image, we can see the height, width, step width, and channel on a DOS text window. Fig.5 shows the height, width, step width, and channel of the image of Fig.4

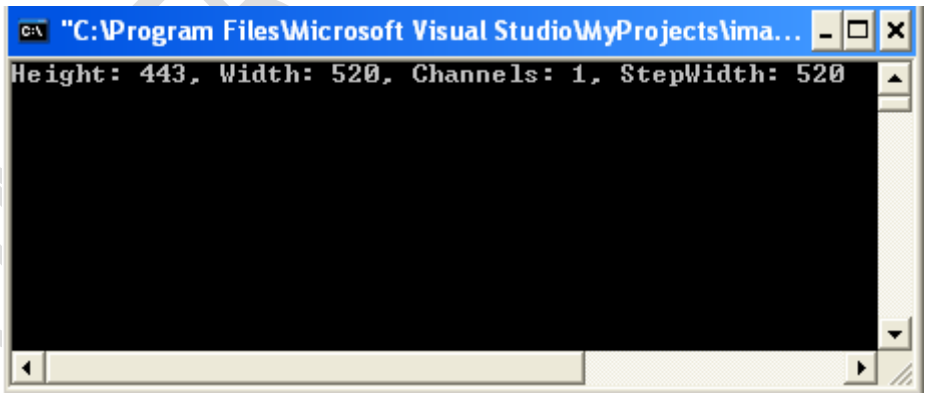


Fig.5: The image data of image 4.1

Step2: For displaying the gray-scale Image completely black and white image with image data we use the image of Fig.4 as an input. After processing this image, we can see the completely black and white image on the specified window. The specified window for this image is "BG". Fig.4.3 shows the completely black and white image of Fig.5

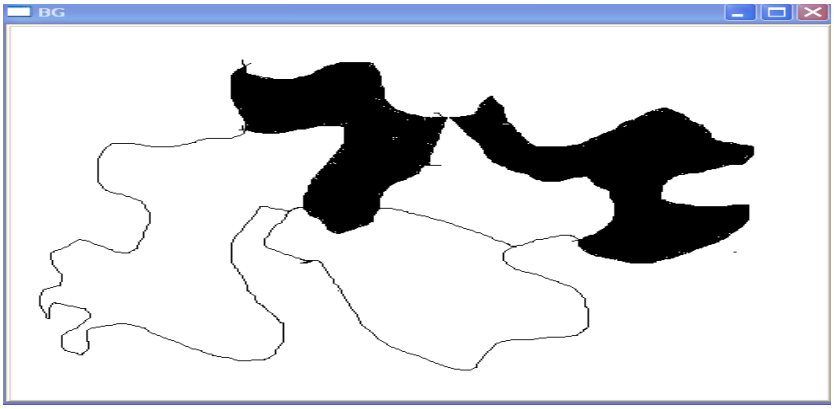


Fig.6: The completely black and white image

Step3: For successful completion of extract the bounded region we use the image of Fig.4 as an input. After processing this image, we can see the specified part of the image on the defined window "BG". We declared the part of the image using function cv_Point (100, 100). Fig.5display the final processed image of Fig.6

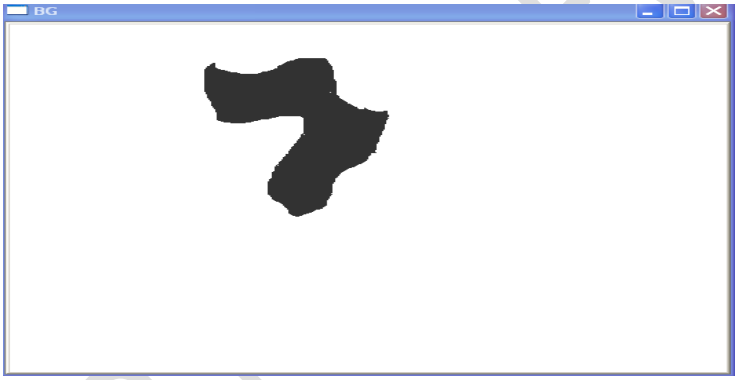


Fig.7 The final processed image

5. Conclusion

In this work, we present an effective region filling algorithm for free regions by using a color scheme to assign different labels to adjoin 4-connected regions. The FF algorithm identifies through all pixels in the image, automatically sets the seeds for flood filling.At first, we use the method of reading the image data and display them on the DOS text window. Then make a gray-scale image completely black. Then select the expected point from the inputted image and give this area a specific color applying thecvFloodFill method. After processing this image, we can see the processed image on the defined window. In our future, work we hope to utilize this concept in the color image.However, we think this algorithm to be competent in giving some probable potential indentation to overcoming the related graphical geometrical matter.

References

[1] C. Pan, S. M. H. Tabatabaei Yazdi, S. K. Tabatabaei, A. G. Hernandez, C. Schroeder, and O. Milenkovic, “Image Processing in DNA,” *bioRxiv*, 2019, doi: 10.1101/2019.12.15.877290.

[2] G. Law, “Quantitative Comparison of Flood Fill and Modified Flood Fill Algorithms,” *Int. J. Comput. Theory Eng.*, vol. 5, no. 3, pp. 503–508, 2013, doi: 10.7763/ijcte.2013.v5.738.

[3] Y. He, T. Hu, and D. Zeng, “Scan-flood fill(SCAFF): An efficient automatic precise region filling algorithm for complicated regions,” *arXiv*, 2019.

[4] B. Kumar, U. K. Tiwari, S. Kumar, V. Tomer, and J. Kalra, “Comparison and Performance Evaluation of Boundary Fill and Flood Fill Algorithm,” *Int. J. Innov. Technol. Explor. Eng.*, vol. 8, no. 12s3, pp. 9–13, 2020, doi: 10.35940/ijitee.l1002.10812s319.

[5] K. Muthukumar, S. Poorani, and S. Sindhu, “Color Image segmentation using Similarity-based Region merging and Flood Fill Algorithm,” vol. 5, no. 06, pp. 40–46, 2016.

[6] J. Esteban, H. Benavides, D. Eduardo, and E. Corredor, “Flood Fill Algorithm Dividing Matrices for Robotic Path Planning,” vol. 13, no. 11, pp. 8862–8870, 2018.

[7] Z. Xiang, “Computer Graphics - Schaum Series.pdf.” p. 342, 2002.